Wintersemester 2021/22
Professor Dr. S. Held
Dr. U. Brenner

Forschungsinstitut für Diskrete Mathematik
Universität Bonn

Einführung in die Diskrete Mathematik
1. Programmierübung

Implementieren Sie Kruskals Algorithmus zur Berechnung eines aufspannenden Baumes mit minimalem Gewicht. Für die Sortierung der Kanten dürfen Sie `std::sort` verwenden. Die Laufzeit für den Rest des Algorithmus soll $O(n + m\alpha(n))$ betragen, wobei n die Zahl der Knoten und m die Zahl der Kanten im gegebenen ungerichteten Graphen ist. Insbesondere sollen Sie also für die Verwaltung der zusammenhangskomponenten die in der Vorlesung beschriebenen Union-Find-Datenstrukturen mit Path-Compression implementieren.

Einlesen der Daten: Dem Programm muss beim Aufruf der Name einer Datei übergeben werden. Ein Aufruf hat also die Form

```
<programmname> <dateiname>
```

Eine gültige Datei, die einen gewichteten Graphen beschreibt, hat das folgende Format:

```
Knotenanzahl
Knoten0a Knoten0b Gewicht0
Knoten1a Knoten1b Gewicht1
...
```

In der ersten Zeile steht eine einzelne natürliche Zahl, welche die Anzahl n der Knoten angibt; Sie können voraussetzen, dass $n < 2^{32}$ ist. Jede weitere Zeile spezifiziert genau eine Kante. Die beiden ersten Einträge dieser Zeilen sind zwei verschiedene nichtnegative ganze Zahlen, welche die Nummern der Endknoten der Kante sind. Dabei nehmen wir an, dass die Knoten von 0 bis $n-1$ durchnummeriert sind. Die Reihenfolge der beiden Knotennummern in einer Zeile spielt keine Rolle, und die Reihenfolge der Kanten kann beliebig sein. Der dritte Eintrag in den Zeilen, die Kanten beschreiben, gibt das Gewicht der Kante an.
Es können parallelen Kanten vorkommen, und der Graph muss nicht zusammenhängend sein. Alle Gewichte sind ganze Zahlen mit Absolutbetrag weniger als 2^{32}.

Ausgabeformat: Das Programm soll eine entsprechende Meldung ausgeben, wenn der Graph unzusammenhängend ist. Ansonsten muss es in der ersten Ausgabezeile das Gewicht des berechneten Baumes ausgegeben. Anschließend sollen die Kanten dieses Baums ausgegeben werden.

b.w.

Beispiel: Eine Eingabedatei für einen Graphen mit fünf Knoten und sechs Kanten kann so aussehen:

```
5
0 1 -5
2 4 6
2 1 3
3 2 -2
1 3 0
3 0 6
```

Die Ausgabe des Programms kann dann so aussehen:

```
The following is a minimum spanning tree, with weight -1.
The graph contains vertices 0,...,4 and the following edges:
{0,1} with weight -5
{3,2} with weight -2
{1,3} with weight 0
{2,4} with weight 6
```

Das Programm muss in C oder C++ geschrieben sein. Es wird empfohlen, C++ zu verwenden. In diesem Fall kann man zum Einlesen und Speichern der Graphen die Klasse `Graph` (eventuell modifizert) aus dem Buch "Algorithmische Mathematik" (Hougardy/Vygen) benutzen. Diese finden Sie (wie alle Programme aus diesem Buch) hier:

`http://www.or.uni-bonn.de/~hougardy/alma/alma2nd.html`

Außerdem dürfen Sie bei Bedarf Teile der C++-Standardbibliothek einbinden. Andere externe Bibliotheken dürfen nicht verwendet werden.
Das Programm muss korrekt arbeiten und ohne Fehlermeldung kompiliert werden können. Der Code muss auf einem gängigen Linuxsystem funktionieren. Achten Sie auch darauf, dass Sie Ihr Programm ausreichend mit Kommentaren versehen.

Testinstanzen finden Sie auf der Homepage der Übungen:
`http://www.or.uni-bonn.de/lectures/ws21/edm_uebung_ws21.html`

Für diese Programmieraufgabe gibt es 20 Punkte.

Abgabe: Der Quelltext des Programms muss bis Dienstag, 16.11.2021, 16:15 Uhr per E-Mail beim jeweiligen Tutor eingegangen sein.